Name:- Purnima kabadwal 1.1

TUTORIAL: 3

3.1

1. Write linear search pseudocode to search an element in a sorted array with maximum comparisons.

```
for (i=0; i<length ;i++)
    {
        if (key ==A[i])
        { swap (A[i] , A[i-1])
            return i-1;
        }
    }
```

2. Pseudocode for iterative and recursive in insertion sort. Insertion sort is called online sort why?

Iterative

```
Void InsertionSort (int i=1; i<n ; i++)
    { j = i-1;
      x = A[i].
    while (j<-1 && A[j]>x)
    {
    A[j+1] = A[j];
        j--;
    }
    A[j+1]=x;
    }
}
```

Recursive Insertion Sort:

```
Void Insertion (int arr[], int n)
   {
     if (n<=1)
        return;

      Insertion (arr, n-1);
      int  last = arr[n-1];
      j = n-2;
      while (j>=0 && arr[j]>last)
        {
           arr[j+1] = arr[j];
             j--;
        }
      arr [j+1] = last;

 }
```

In Insertion sort, we give input one by and place each one at right order with comparision from already traced element we need the whole array simultaneous that's why its called an online Sorting algorithm.

3. Complexity of all sorting algorithms. 3.2

	Best Case	Worst Cast
Bubble Sort	$O(n)$	$O(n^2)$
Selection Sort	$O(n^2)$	$O(n^2)$
Insertion Sort	$O(n)$	$O(n^2)$
Quick sort	$O(n \log n)$	$O(n^2)$
Merge Sort	$O(n \log n)$	$O(n \log n)$
Count Sort	$O(n+k)$	$O(n+k)$

4. Divide all sorting algorithms into inplace/stable/online sorting.

Online sorting :- Insertion Sort (Partial)

Stable Sorting :- Merge sort, Insertion sort

Bubble Sorting :-

Bubble sort, Inserting sort, Selection sort.

Name: Purnima Kabadwal

5. Write iterative / recursive pseudo code for binary search. What is the time and space complexity of linear and Binary search.

Iterative Binary Search

```
        while (low <= high)
    {  int mid = (low + high) /2 ;
         if (arr[mid] == key)
           return mid.
      else  if (arr [mid] > key)
        high = mid-1 ;
       else
        low = mid+1 ;
       }
```

Iterate Binary search

T.C :- $O(\log n)$

S.C :- $O(1)$

Recursive Binary

Time comparting :- $O(\log n)$

Space Compbexing :- $O(\log n)$.

Linear search ↦

Time complexity :- $O(n)$

space complenty :- $O(1)$

3.3

6. write recurrence relation for binary search

$$T(n) = T(n/2) + 1$$

$$T(1) = 1$$

7 Find two indexed such that $A[i] + A[j] = k$ in ~~index~~ minimum time compeexity.

```
int find-sumPair(int A[], int n,int k)
  {
      Sort (A,n);
       i=0.
       j=n-1;

    while (i<j)
     { if (A[i ] +A[j ] = =k)
           return 1;
        else if (A[i ] + A[j] <k)
           i++;
           else
             j--;
        }
       return -1;
   }
```

8. Which sorting is best for practical use? Explain.

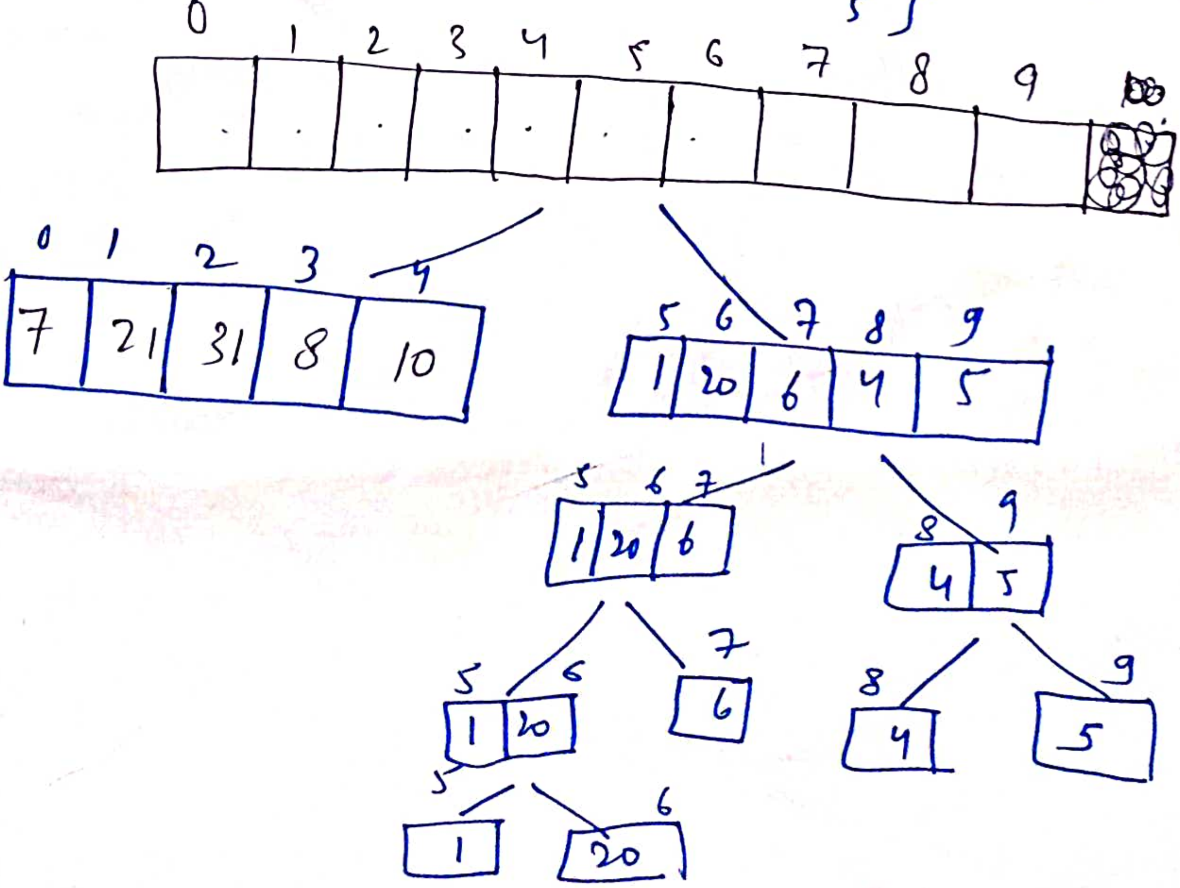
Quick sort is the fastest general purpose sort. In most practical situations quick-sort is a method of choise.

If stability is important and space is availble, merge sort might be best.

9. What do you mean by inverstions?

Invesrsion count of an array suggests how close array is from being sorted.

Given :- ~~A~~array arr[] = {7, 21, 31, 8, 10, 1, 20, 6, 4, 5}



10 In which cases Quick sort will give best and worst time compleixity. 3.4

Worst case: Time complexity of Quicksort is $O(n^2)$ it occurs when picked pivot is always an extreme (smallest or largest) element. This happens when input array is sorted or reverse sorted and either first or last element is picked.

Best Case:- Best case time complexity of Quick sort is $O(\log n)$ it occurs when pivot element is middle element always for the partition process.

11. Quick SORT

Worst Case :-

T(0) :-

$T(0) = T(1) = 0$

$T(N) = N + T(N-1)$

$\rightarrow O(n^2)$

Best Case :-

$T(0) = T(1) = 0$

$T(N) = 2T(n/2) + N$

$\rightarrow O(n \log n)$

Merge SORT

$T(n) = 2T(n/2) + n$

Basis	Quick sort	Merge sort
Portion	splitting is done in any ratio.	Array is partition in just 2 halves
Works well on	Smaller array	Fine on any size Of array.
Addition space	Less (inplace)	More(not inplace
Sorting method	Internal	Eternal
Stability	unstable	Stable

12. Selection sort isn't stable by default but can you write a stable ~~[illegible]~~ version of selection sort.

Selection sort can be made stable if instead of swapping, minimum elements is placed in its position without swapping i.e by placing the number in its portion by pushing every element one step forward.

13. Bubble sort scans whole array even when array is sorted. Can you modify bubble sort.

N kabadwae 5.1

Modified Bubble sort :- 3.5

```
void Bubble sort (int A[], int n)
{
  int flag;
  for (int i=0; i<n-1; i++) {
      flag =0;
   for (j=0 ;j <n-1 -i ;j++)
    {
        if (A[j] >A[j+1])
         {
           swap (A[j], A[j+1];
          flag =1;
         }
    }
  if (flag ==0)
   break;
  }
}
```